2021 IEEE 37th International Conference on Data Engineering (ICDE) | 978-1-7281-9184-3/20/$31.00 ©2021 IEEE | DOI: 10.1109/ICDE51399.2021.00284

The IoT Meta-Control Firewall

Soteris Constantinou*, Andreas Konstantinidis‡*, Demetrios Zeinalipour-Yazti* and Panos K. Chrysanthis†,*

* Department of Computer Science, University of Cyprus, 1678 Nicosia, Cyprus
‡ Department of Computer Science & Engineering, Frederick University, 1036 Nicosia, Cyprus
† Department of Computer Science, University of Pittsburgh, PA 15260, USA
{constantinou.sotiris, akonstan, dzeina}@cs.ucy.ac.cy, panos@cs.pitt.edu

***Abstract*— Internet of Things (IoT) devices have penetrated massively into smart environments (e.g., smart-homes, smart-cars or more generally smart-anything). Besides data collection, many IoT devices also enable the execution of Rule Automation Workflows (RAW), which span from simple predicate statements to procedural workflows capturing a smart actuation pipeline. RAW aim to meet the convenience (comfort) level of users under specific conditions (e.g., raise room temperature to 22°C if cold), but unfortunately cannot express long-term objectives of users (e.g., consume less than 400 kWh in December). In this paper, we present an innovative system, coined IoT Meta-Control Firewall (IMCF), which internally deploys an AI-inspired Energy-Planner (EP) algorithm that exploits domain-specific operators to balance the trade-off between convenience and energy consumption in satisfying the RAW pipelines of users. IMCF filters the RAW pipelines in a way that these do not conflict with the long-term objectives of users (like a network firewall). Our experimental evaluation with extensive real traces from an apartment, a house, and campus dorms shows that IMCF achieves very high levels of user convenience while remaining within the target energy consumption budgets expressed by users.**

I. Introduction

Internet of Things (IoT) refers to a large number of physical devices being connected to the Internet that are able to "see", "hear", "think", "react", perform tasks, as well as communicate with each other using open protocols [1], [2], [3], [4]. IoT enables the development of smart applications in various domains, such as transportation, healthcare, industrial automation, emergency response and business, having significant impact on the quality of people's life, the growth of the world's economy, and security [3]. IoT data management is becoming a very hot topic in data engineering and we overview this movement in the related work. Studies showed that a typical family in the developed world owns about 5-10 internet-connected devices, such as smartphones, smartTVs, and smart-home devices. According to Gartner[1] it is expected to increase to more than 500 smart devices by 2022.

Besides data collection, many IoT devices also enable the execution of *Rule Automation Workflows (RAW)*, which span from simple predicate statements to procedural workflows capturing a smart actuation pipeline in tools like Apilio.io, Apple Automation or IFTTT, which controls Philips Hue lights, BMW i3 EVs or Daikin A/C units [5], [6]. RAW aim to meet the convenience level of users under specific conditions (e.g., *"warm house to 22°C if cold or preheat Electric Vehicle when approaching"*). In the simplest case, a user expresses preferences manually through a vendor-specific smartphone app or an integrated app (e.g., see Fig. 1 left). This process requires continuous attention by custodians, making it a cumbersome process that generates erroneous executions and that clearly calls for more automated approaches.

[1]Gartner Inc., `http://tiny.cc/4gk8tz`

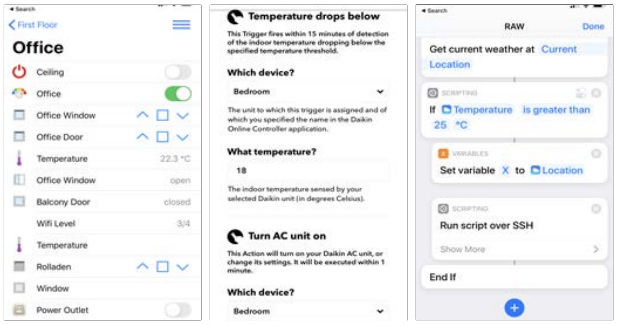

Fig. 1. **Rule Automation Workflow (RAW) management:** by manual means (openHAB, left), by predicate rules (IFTTT, center) and by procedural rules (Apple Automation, right). None of these supports the expression of long-term objectives (e.g., energy consumption) we propose in this work, which work complementary to custom rules.

One of the most straightforward approaches to achieve a smarter RAW is to adopt the so-called *trigger-action* model. Users control the behavior of an IoT device by specifying triggers (e.g., *"if it is sunny outside"*) and their resultant actions (e.g., *"turn off the lights"*). Because of its conceptual simplicity, the trigger-action model (a.k.a. Event-Condition-Action) has attracted significant attention with ifttt.com (*"If This Then That"*) becoming one of the first large-scale deployments (see Fig. 1 center). Services like Apilio expanded the expressiveness of the RAW with Boolean predicates (e.g., conjunctions) and Apple Automation even introduced procedural programming constructs, like variables, while loops, if statements and functions (see Fig. 1 right) to bring RAW smart actuations to new levels.

However, none of the above RAW technologies enables individuals or group of users to express their convenience (comfort) preferences while achieving some long-term objective in a convenient manner.

In our context, the **long-term objective** relates to *energy consumption* (e.g., in kWh), which is motivated by Euro-

978-1-7281-9184-3/21/$31.00 ©2021 IEEE
DOI 10.1109/ICDE51399.2021.00284

pean's Commission calls for a climate-neutral Europe by 2050[2]. Unfortunately, the CO_2 pollution of ICT (production of devices and usage) is expected to rise from 4% to 8% by 2025[3]. We claim that by consuming energy more intelligently (i.e., using smart actuations) can greatly contribute to the environmental impact of ICT, enabling us to improve living conditions and also respect the environment reaching agreed targets. To understand this desideratum, consider two separate examples, one in a smart-home and the other one in smart-dormitory setting:

A. Motivational Scenarios

Smart-Home: As a first example consider a single family that has invested in photovoltaic technology to cover its heating, mobility and other energy requirements. In our scenario, the family has a yearly budget of 8500 kWh (i.e., yearly production of this household under a net-metering scheme, where energy excess on a sunny day can be used at later stages within a yearly cycle) and aims to spend this energy budget through RAW that configures the energy consumption preferences of the family (e.g., room temperatures across the year in the house as well as auxiliary lighting). The family is willing to adapt its desired interior temperature preferences (e.g., adapting indoor temperature by 1°C, one can save 6% of its energy consumption[4]) according to production and consumption patterns, but has no clue how the RAW pipelines contribute to the target of using only 8500 kWh per year. Currently, they rely on manual guess-work and manual planning that is cumbersome and error-prone [7] (Section IV overviews the related work showing that no other solution is available to this problem).

Smart-Dorms: As a second example consider the SAVES [8] project, which was an inter-dormitory energy-saving competition within the framework of the European Commission Intelligent Energy – Europe (IEE) programme that took place between 2014-2016. The project aimed to instill energy-saving habits to students at a key moment of change in their lives so that they can continue energy-saving actions throughout their entire lives. SAVES aimed at delivering 8% average electricity savings in participating dormitories. Students at the University of Cyprus participated with great excitement and passion that eventually led to a saving of 4,44%. Even though students applied common sense and perseverance in achieving the energy reduction target, there was a lack of intelligent control to reach the higher desired target.

B. The IMCF Approach: Overview

In this paper, we present an innovative system, coined the *IoT Meta-Control Firewall (IMCF)*, which aims to fill the gap of manual RAW tuning to reach the energy consumption targets. The user (or group of users) starts out by defining a vector of RAW rules, dubbed Meta-Rule-Table (MRT),

[2]EU 2050 long-term strategy, https://tiny.cc/9wu8iz
[3]DW, https://p.dw.com/p/3Lvxs
[4]U.S. Dept. of Energy, https://tiny.cc/kwfijz

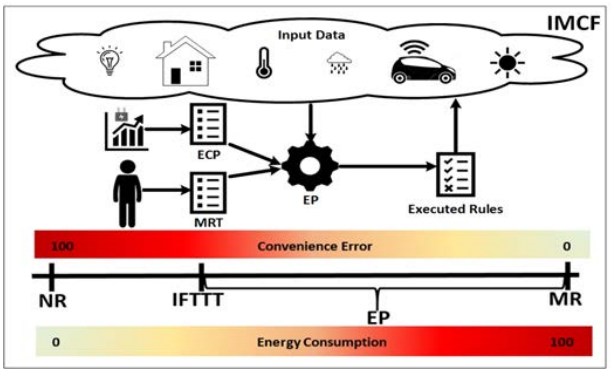

Fig. 2. The *Energy Planner (EP)* algorithm proposed in this work is an AI-inspired algorithm that finds the best possible energy consumption strategy with respect to user convenience by only using a *Meta-Rule-Table (MRT)* profile and *Energy Consumption Profile (ECP)* and without the necessity of a learning history used by machine learning methods.

and an *Energy Consumption Profile* (Table I), dubbed ECP (see Fig. 2). The high-level objective is to identify among all MRT rules the ones that must be dropped so that the user stays within the desired energy budget according to the *ECP* history. The IMCF automatically adopts an intelligent energy amortization process combined with an AI-inspired algorithm we propose, to balance the trade-off between convenience and energy consumption while managing RAW pipelines of users. Particularly, it utilizes an intelligent search algorithm, called *EP (Energy Planner)*, which goes over the exponentially large search space of $\sum_{r \leq N} r$-combinations (where $N = |MRT|$), yielding quickly the rules to be dropped. IMCF adapts the RAW pipelines in a way that these do not conflict with the long-term objectives of users (by dropping certain rules based on preference priority).

The RAW pipelines are distinguished in our discussion into the *convenience* and *necessity*. The *convenience rules* aim at promoting an individual's physical comfort (e.g., room temperature, ambient lighting, pre-heating of car, operation readiness of general appliances or whatever is considered tentative comfort), while the *necessity rules* are those rules that should always be executed regardless of whether the long-term target is met. Without loss of generality and for ease of exposition, consider only the convenience rules, sorted in order of importance, for the remainder of this work.

In respect to processing the RAW rules, one could ignore the RAW rules completely, obtaining in this way the best energy consumption but the worst convenience (we call this the *No Rule (NR)* method - see Fig. 2). In contrast, a user could obtain maximum convenience by having every single preference rule inside RAW executed that would obviously bring the highest convenience but at the same time it would consume the highest amount of energy (we call this the *Meta-Rule (MR)* method). The IFTTT approach, in the absence of a detailed user preference profile, being an arbitrary sequence of rule executions would then be somewhere in between these two borderline cases, while EP is a more rigorous optimization method of arbitrary rules.

TABLE I
ENERGY CONSUMPTION PROFILE (ECP) OF FLAT MODEL (USED IN THE EVALUATION)

Months	kWh per month	kWh per hour
January	775.50	1.04
February	528.75	0.71
March	246.75	0.33
April	141.00	0.19
May	176.25	0.24
June	211.50	0.28
July	246.75	0.33
August	317.25	0.43
September	211.50	0.28
October	176.25	0.24
November	211.50	0.28
December	423.00	0.57
Total	3666.00	-

A Meta-Rule-Table (MRT) satisfies a user's preference rules along with the long-term energy objective. Consider for example the following constraint/meta-rule: *"Keep the monthly energy consumption budget below 100 euro"* (1 kWh costs around 0.20 Euros in EU[5], so monetary to energy conversion can be carried out directly). The incorporation of multiple rules may cause several deficiencies, such as rules competing or throwing a clash with each other, rules becoming infeasible to be satisfied and/or rules that their behavior depends on the output of other rules. This is mainly due to the complexity of current controllers to autonomously track and monitor a high number of rules that may be set by the user in different periods, under different circumstances [9]. For example, the above meta-rule that refers to the monthly energy budget not exceeding 100 euro, will conflict with another meta-rule that turns the AC to cooling when the room temperature is $> 18°C$ and the monthly energy budget is already consumed. Similarly, the above meta-rule may also cause a conflict with a meta-rule that turns on the lights when the season is winter, and the time is between 13:00-16:00.

Our intelligent algorithm enables some user to find an energy-efficient plan for the execution of a set of convenience rules encoded in MRT and a tentative ECP, satisfying several objectives subject to a specific energy constraint. The efficiency of the proposed techniques is measured by the following metrics: (i) the *Convenience Error* (F_{CE}); and (ii) the *Energy Consumption* (F_E) required for finding a near-optimal plan of meta-rules. In summary, in this paper we have the following contributions:

- We propose a novel notion of filtering RAW workflows and formally defined. In this scope, we propose the design and implementation of the Energy Plan algorithm within an IoT Meta-Control Firewall (IMCF) that has the ability to handle the user's convenience profile by considering the user's energy budget.
- We present a complete system architecture of our IMCF smart energy management system implemented inside the openHAB stack.
- We evaluate our design with extensive experimentation on real datasets with anonymized measurements from a real residential apartment that comprises of a variety of sensors and approximately 5M readings (1.09 GB in total), showing that EP can be premise for energy-aware smart actuations in the future. We finally also demonstrate the utility of our prototype system.

[5]EU Statistics, http://tiny.cc/53vijz

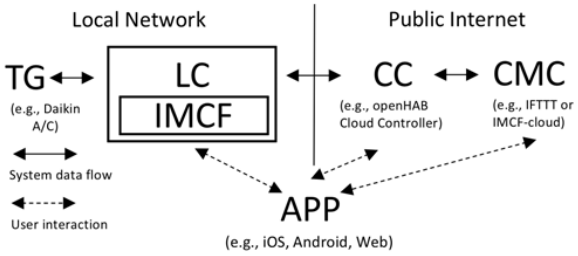

Fig. 3. Overview of the IMCF System Architecture.

The remainder of the article is organized as follows: Section II presents our proposed framework's system architecture and its internal components. Section III presents our experimental methodology and results. Section IV provides other related work while Section V concludes the article.

II. THE IOT META-CONTROL FIREWALL (IMCF) SYSTEM ARCHITECTURE

In this section, we describe a prototype system we have developed, called The IoT Meta-Control Firewall (IMCF) system[6]. The IMCF has been implemented using the open Home Automation Bus (OpenHAB) [10], the Linux crontab daemon, as well as the Laravel PHP web framework following the model–view–controller architectural pattern. We start out with a discussion of the system architecture, followed by the IMCF algorithm, and then describe the GUI we have developed. The GUI integrates directly into OpenHAB's mobile and web Panel view for both interactive management of IoT and automated management of Energy-aware MRT pipelines using the EP described in this work.

A. System Architecture

Our system architecture comprises of the following components: (i) a full-fledge local controller implemented inside the openHAB stack, which is a smart home management software; and (ii) IMCF, which is the software system that encapsulates the complete application logic of the energy management stack we propose along with the respective user interfaces.

Local Controller (LC): is a java-based system installed on a micro device, like a Linux Raspberry PI, running on the local network of a user. The LC will be in direct communication with the IoT devices (i.e., *Things (TG)*) to instruct them based on the preferences registered by a user (see Fig. 3). A user will typically download the openHAB smartphone *application (APP)*, for iOS or Android, and interact with TG through

[6]The IoT Meta-Control Firewall, https://imcf.cs.ucy.ac.cy/

LC. For the implementation of LC we decided to extend the openHAB stack, which is a vendor and technology agnostic open source automation software for smart home that provides a rich ecosystem of bridges through which a user can interact directly with IoT devices (e.g., Daikin Smart A/C, Phillips HUE lights) both locally and remotely. This gives us the benefit to achieve maximum IoT market compatibility as the integration of IoT is always an immense challenge.

To realize the operation of LC, consider for example a user inside his smart space that uses an APP to increase the temperature of an A/C from 21 to 25 degrees Celsius (see Figure 5a-b). This manual interaction goes directly to LC that eventually communicates with TG (on older units this is typically unencrypted http communication channels, either http querystring or in some cases JSON web 2.0 interactions). When a user's APP is outside a smart space, the network firewall and Network Address Translation (NAT) will obviously not let this user interact with LC. As such, the user's APP connects to the *Cloud Controller (CC)*, which is a server on the public Internet that communicates and controls LC remotely.

The complete picture can tentatively be complemented by a *Cloud Meta-Controller (CMC)*, like IFTTT [6], which can enable the user to configure and run various custom rules. CMC would in this case interact with CC that would in turn interact with LC that would eventually interact with TG, all under the *manual* control of the user APP.

The IMCF Component: is a software extension to LC we have implemented to enable the adaptation of convenience preferences to meet the long-term energy planning targets of individuals or group of individuals. It has been developed in a way that encapsulates the implementation of the EP algorithm but also the GUI and storage necessary to allow the user interact with the system. The EP algorithm is implemented as a JAVA library which takes the user configurations from a local MariaDB persistency layer. The storage layer is populated by the user using the APP, which has been configured in a way to integrate seamlessly the MRT rule definition process through a web-based GUI (see Figure 5c,d).

The GUI code is written in the Laravel PHP web framework following the model–view–controller architectural pattern as well as JavaScript and HTML. Our complete code is approximately 2500 lines-of-code plus 3000 going to the GUI. For the GUI code execution, we rely directly on the NGINX webserver available on Raspberry PI [11], while for the IMCF EP library we invoke the cron job daemon that reliably executes the EP every few minutes. In case devices have to be turned on or off, the IMCF system has the following options in our system:

- *Binding-mode*, where IMCF exploits the rich ecosystem of bridges available on the openHAB open source project to interact with local devices. We use this as the default mode, as it allows our platform to scale to a very wide spectrum of IoT devices.

  Example [7]**:**
  daikin.things: `daikin:ac_unit:living_room_ac [ host="192.168.0.5" ]`
  daikin.items: `Switch DaikinACUnit_Power channel="daikin:ac_unit:living_room_ac:power" Number:Temperature DaikinACUnit_SetPoint channel="daikin:ac_unit:living_room_ac:settemp"`
- *Extended mode*, where IMCF implements locally the custom instructions for enabling and disabling the various TG devices in the smart space of a user. An example of this mode is the following command:
  Example: Setting Daikin in Cool Mode 25 degrees [8].
  `http://192.168.0.5/aircon/set_control_info?pow=1&mode=3&stemp=25&shum=0`

  Given that many of the IoT communications are unencrypted, this can easily be captured by deep packet analyzers like Wireshark. Additionally, in order to avoid any additional CMC, CC or LC interactions with the Daikin TG, we also configure the LC network firewall with the `iptables` command to disable TCP flows to designated TG devices on the local network. In this case, IMCF works actually like a real network firewall by blocking all outgoing traffic from LC to TG.
  Example:
  `iptables -A OUTPUT -s 192.168.0.5 -j DROP`

B. The $IMCF$ algorithm

The $IMCF$ algorithm is composed of two subroutines: (i) the *Amortization Plan (AP)*; and the (ii) the *Energy Plan (EP)*. The amortization plan is responsible for calculating the maximum energy budget constraint (E_p) through a preselected amortization formula. Then an artificial intelligence approach is executed every t seconds (e.g., hourly, daily, monthly, yearly preference) over a time period p (i.e., the complete duration of the execution) for generating an energy plan solution s^* for optimizing the Convenience Error

$$\min F_{CE} = \sum_{k=1}^{t}(\frac{1}{N}\sum_{i=1}^{N}\sum_{j=1}^{D} ce_j(MR_i)), \quad (1)$$

where ce_j is the difference between the desired output value $\Omega_i^j \in \Re$ of a rule set by a user (temperature or light intensity level) and the actual value $O_i^j \in \Re$ set by the controller, given by: $ce = |\Omega_i^j| - |O_i^j|$.
Subject to satisfying the Energy Consumption $F_E(s^*) \leq E_p$, where:

$$F_E = \sum_{k=1}^{t}(\frac{1}{N}\sum_{i=1}^{N}\sum_{j=1}^{D} e_j(MR_i)), \quad (2)$$

E_p is total available energy budget for the complete period p during which the execution of our algorithm takes place, N is the total number of meta-rules, D the set of all IoT devices

[7]OpenHAB Daikin Binding, `http://tiny.cc/6jk8tz`
[8]Daikin Control, `https://github.com/ael-code/daikin-control`

and e_j is the energy consumption of device j given the action defined by output O_i^j of meta-rule MR_i, given by:

$$E = \begin{cases} e_j, & \text{if } O_i^j \text{ is executed} \\ 0, & \text{otherwise} \end{cases},$$

where e_j is the energy cost of device j for MR_i.

In this paper, we have adopted a hill-climbing algorithm, an iterative local search heuristic, which doesn't require a learning history (like respective Machine Learning techniques), does not require a target function (e.g., like A*) and is straightforward to be implemented in a resource-constraint setting like local smart controllers (e.g., Raspberry).

Amortization Plan Algorithm: The $AP()$ subroutine is initially executed for calculating the energy budget constraint E_p, subject to a monthly residence *Energy Consumption Profile* ECP. There are several amortization strategies that can be used, such as the following:

(i) Linear Amortization Formula (LAF): In this case, the total energy consumption TE can be linearly allocated throughout a pre-specified period p of duration time t, which can be set as yearly, monthly, daily, hourly and so on, giving the energy budget constraint:

$$E_p = \frac{TE}{t}, \tag{3}$$

where TE is the total energy allocated for the complete period p. In our ECP example of Table I, the flat consumes a total energy $TE = 3666$ kWh yearly, on average. In this case, if an hourly energy budget period is selected by the user, then the energy budget constraint E_h will be calculated as $E_h = 3666/8928 = 0.742$ kWh, for a duration $t = 12 \times 31 \times 24 = 8928$, indicating the hourly available budget for the whole year.

(ii) Balloon Linear Amortization Formula (BLAF): In this case, the user saves a percentage π of energy from total energy TE for a period of time $\lambda < t$, the so-called balloon σ, which is used in the remaining period $\lambda' = t - \lambda$ that the energy consumption is higher. The energy budget constraint E_p for a period p of duration t is calculated as follows:

$$E_p = \begin{cases} \frac{TE}{t} - \frac{\sigma}{\lambda}, \text{for } \lambda \text{ period} \\ \frac{TE}{t} + \frac{\sigma}{\lambda}, \text{for } \lambda' \text{ period} \end{cases}, \quad \text{where } \sigma = (\frac{TE}{t} \times \lambda) \times \pi. \tag{4}$$

In our example, if the user desires to save $\pi = 30\%$ of the total energy consumption $TE = 3666$ kWh, for $\lambda = 7$ months (e.g., for April to October) that the consumption is lower than the remaining $\lambda' = 5$ months (i.e., November to March) then $\sigma = (305.5 \times 7) \times 0.3 = 641.55$ kWh. Therefore, the energy consumption for seven months, between April to October, will be $E_p = 397.15$ and for five months, between November to March will be $E_p = 213.85$ kW. The corresponding hourly energy budget constraint of this formula will be $E_h = 397.15/(31 \times 24) = 0.53$ kWh and $E_h = 213.85/(31 \times 24) = 0.28$ kWh, accordingly.

Algorithm 1 $IMCF$: generates an energy-efficient plan

Input: MRT: Meta-Rule Table; k: components to be modified; τ_{max}: max iterations; t: time granularity; apl: amortization plan; ECP: Energy Consumption Profile
Output: An energy plan solution $s^* = (s_1, \ldots, s_N)$

AP(apl, p, ECP) ▷ **Amortization Plan Routine**
  switch (apl)
  a: $E_p \leftarrow LAF(t, ECP)$ ▷ use linear Eq. (3)
  b: $E_p \leftarrow BLAF(t, ECP)$ ▷ use balloon Eq. (4)
  c: $E_p \leftarrow EAF(t, ECP)$ ▷ use ECP-based Eq. (5)

EP($MRT, k, \tau_{max}, i, E_p$) ▷ **Energy Plan Routine**
  $s^* \leftarrow init_i(MRT)$ ▷ s^*: initial solution for time i
  $(F_E, F_{CE}) \leftarrow evaluate(s^*)$ ▷ with Equations (1),(2)
  While $\tau < \tau_{max}$ **do** ▷ τ: current iteration
    $s \leftarrow optimization(s^*)$ ▷ randomly select k positions and swap their binary value
  $(F_E, F_{CE}) \leftarrow evaluate(s)$ ▷ with Equations (1),(2)
    If ($F_E(s) \leq E_p$) && ($F_{CE}(s) < F_C(s^*)$) **then**
      $s^* \leftarrow s$ ▷ Set s as the current solution s^*
    EndIf
    $\tau++$ ▷ Increase iterations
  EndWhile
  return s^* ▷ Return the final energy plan solution

$apl \leftarrow AP(apl, t, ECP)$;
return ($\forall_i^t\ EP(MRT, k, \tau_{max}, i, apl)$)

(iii) ECP-based Amortization Formula (EAF): In this case, a set of weights is calculated using the *Energy Consumption Profile* ECP vector (e.g., see Table I). The weights are then used to define the energy budget constraints for a user-defined period over an available energy budget E:

$$E_p = \{\frac{w_i \times E}{t/|ECP|}\}, for \quad i = 1, \ldots, |ECP|, \quad \text{where } w_i = \frac{TE}{ECP_i} \text{ and } \sum_{i=1}^{|ECP|} w_i = 1, \tag{5}$$

TE is the total energy consumption derived from the ECP, E is the user-specified available energy budget, $|ECP|$ is the size of the *Energy Consumption Profile* vector and $t/|ECP|$ normalizes the energy budget based on the time granularity duration t. Clearly, t could have taken a different granularity (e.g., day, hour or even minute), given that this information is typically available in energy monitoring systems.

For example, lets assume an hourly energy budget period and an available yearly budget $E = 3500$ kWh selected by a user of a flat with an ECP indicated in the left column of Table I. The total energy consumption derived from the ECP set is $TE = 3666$ kWh and $|ECP| = 12$. Therefore $w_1 = 0.211, w_2 = 0.144$, and so on until $w_{12} = 0.115$. The hourly energy consumption per month can be calculated as $\{\frac{w_i \times 3500}{31 \times 24}\}$.

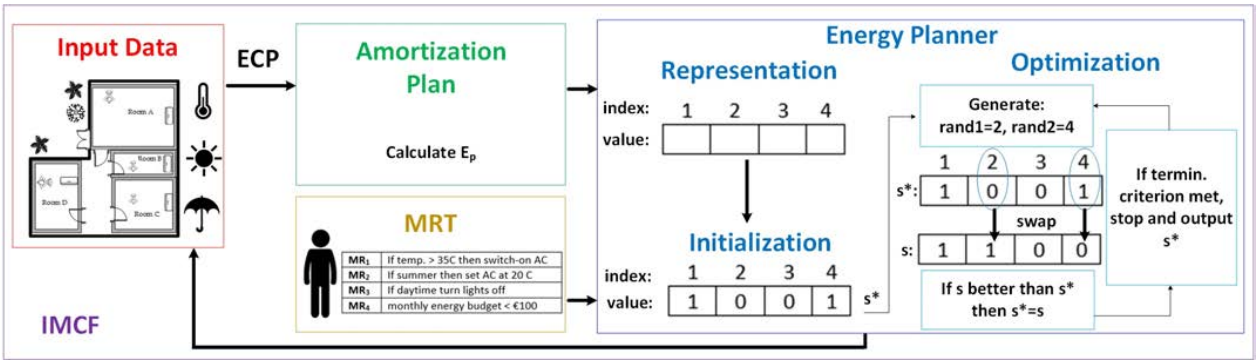

Fig. 4. Example execution of the IMCF framework Planner.

Energy Plan Algorithm: In this subsection, we discuss our EP algorithm and its related components and parameters.

Solution Representation: An energy plan solution is a vector $s =< s_1, \ldots, s_N >$ of size $N = |MRT|$. A vector component s_i represents a meta-rule in table MRT, where $s_i = 0$ means ignoring meta-rule at position i of table MRT and $s_i = 1$ means adopting meta-rule at position i.

Initialization: At the beginning of the local search heuristic an initial solution s^* is developed in line 8 that will specify the initial state of the algorithm. An initial solution can be generated randomly or deterministically. In the latter, a deterministic solution for the EP can be to set all vector components to 1, meaning that all meta-rules will be greedily triggered, favoring in this way the convenience error objective, but having a high probability of violating the E_p constraint. In the case of a random initialization, the values of all vector components are uniformly randomly selected.

Optimization: For the optimization step, a hill-climbing local search heuristic is utilized for local optimization with neighborhoods that involve changing up to k components of the solution, which is often referred to as k-opt. During the optimization process k components are uniformly randomly selected and their binary value is swapped. Here it is important to note that any heuristic or meta-heuristic approach can be utilized in the EP optimization step.

Evaluation: Each solution s is evaluated using the performance metrics F_E and F_{CE} of Equations (1) and (2) in lines 9 and 12. A solution s is considered better and replaces the current best solution s^* if $(F_E(s) \leq E_p)$ && $(F_{CE}(s) < F_{CE}(s^*))$.

Termination criterion: the energy planner stops when τ_{max} iterations are completed. Alternatively, the algorithm can iterate until $\nexists s | F_{CE}(s) < F_{CE}(s^*)$. However, in the absence of any knowledge on the optimal solution this may result in an infinite loop.

Case scenario: Consider the simplified scenario of Fig. 4 in which a user sets four meta-rules in the MRT for a four-room residence, which along with some input data from the house's sensors as well as some online web services (e.g., weather forecasting website) are forwarded to the $IMCF$. $IMCF$ initially runs the amortization plan subroutine using a pre-selected amortization formula as well as the *Energy Consumption Profile* ECP and calculates an energy budget constraint E_p. Then it converts the MRT to a binary vector, in which each position of the vector represents a meta-rule in the MRT. A random initialization process generates the first solution $s^* =< 1, 0, 0, 1 >$, which means that meta-rules 1 and 4 will be triggered and meta-rules 2 and 3 will be ignored. Solution s^* is evaluated using the performance metrics of Energy Consumption and Convenience Error. During the optimization, $k = 2$ vector components are modified using a uniform random generator. In this example, the value of vector component 2 is swapped from 0 to 1 and the value of component 4 is swapped from 1 to 0. The newly generated solution $s =< 1, 1, 0, 0 >$ is again evaluated and compared with the current best solution s^*. At each iteration, when s is better than s^* then s becomes the s^*. The algorithm stops when the termination criterion is met.

C. Discussion and Analysis

In this subsection we qualitatively compare our proposition to the baselines and also carry out a respective analysis.

Baseline Approaches:

(i) No-Rule (NR): the first baseline approach ignores all rules in the Meta-Rule-Table (see Table II used in the flat dataset, the rest datasets later use uniformly random variations of the same table) and does not modify the behavior of the autonomous devices, thus this contradicts with the user's convenience. F_E is obviously always 0 since no IoT device is turned on. On the other hand, F_{CE} is measured as a percentage of convenience a user would have if that user executed all rules and F_T is only the cost of doing that calculation. Consequently, the energy consumption of this approach is minimum and the convenience error is maximum.

(ii) Meta-Rule (MR): the second baseline approach ignores the energy consumption and executes all rules (greedily) in the Meta-Rule-Table for satisfying all meta-rules set by the user (again, see Table II used in the flat dataset). Consequently, the energy consumption of this approach is maximum and the convenience error is minimum as IoT will operate maximally.

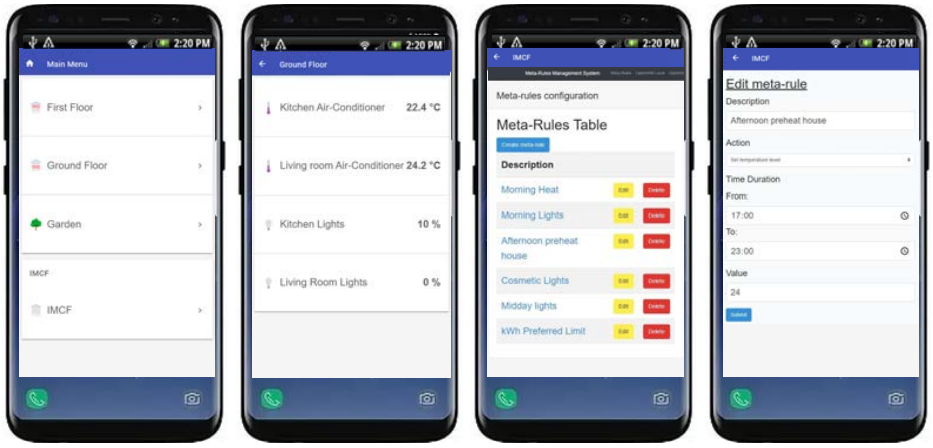

Fig. 5. **IMCF Graphical User Interface:** Integration of the IMCF Software Library in the openHAB Home Automation Stack. From left to right: (a) Interactive and Automated Menu; (b) Dashboard for smart space current state; (c) Meta-Rule-Table Configurator; and d) MRT data entry form.

Performance Analysis: We analytically derive the performance of $IMCF$ with respect to the estimated convenience error CE and energy consumption E. We adopt a worst-case analysis as it provides a bound for all input considering the two extreme methods, i.e., No-Rule and Meta-Rule, as explained in Section I. Our experimental evaluation in Section III, shows that under realistic and real datasets our approach performs more efficiently than the projected worst case. The analysis ignores any energy not directly associated with the meta-rules table MRT.

Lemma 1. *Our IMCF approach has a convenience error of* $F_{CE} = \frac{1}{n}\sum_{i=1}^{D}\sum_{j} ce_j(MR_i), i = 1, \ldots, n$, *where* $n > 0$ *is the number of meta-rules that will be executed.*

Proof. The energy planner will select at least $n > 0$ meta-rules to be executed satisfying in this way the energy budget constraint. In the worst case where the energy budget is equal to zero, IMCF will act as the NR approach providing $F_{CE} = 1$. On the other hand, the MR approach by greedily executing all meta-rules in the MRT will offer an $F_{CE} = 0\square$

Lemma 2. *Our IMCF approach has an energy consumption of* $F_E = \frac{1}{n}\sum_{i=1}^{D}\sum_{j} e_j(MR_i), i = 1, \ldots, n$, *where* $n \leq N$ *is the number of meta-rules that will be executed.*

Proof. The energy planner will select at most $n \leq N$ meta-rules to be executed satisfying in this way the energy budget constraint. In the worst case scenario where there is no energy budget constraint, IMCF will act as the MR approach providing $F_E = 1$. On the other hand, the NR approach by not executing any meta-rule of the MRT will offer an $F_E = 0\square$

D. Graphical User Interface (GUI)

Our prototype GUI provides all the functionalities for a user participating in IMCF. The GUI is divided into a Meta-Rule-Table interface and the OpenHAB Rules Table, respectively, as shown in Figure 5d. The Meta-Rules interface prompts users to define kWh preferred limits, temperature and light values for any configured time slots. The OpenHAB Rules Table records are retrieved through the OpenHAB Rest API system that consists of smart device sensor measurements installed and pre-configured in a building. These rule combinations are used by the EP algorithm.

At a high level, our GUI enables the following functions: (i) record OpenHAB item measurements/values on local storage and present those on a table; (ii) configure various meta-rules in regards of kWh limit, temperature and light values; (iii) operate the IMCF framework and get an efficient execution considering user satisfaction along with balanced Convenience Error (F_{CE}) and Energy Consumption (F_E).

III. Experimental Methodology & Evaluation

This section presents an experimental evaluation of our proposed framework. We start-out with the experimental methodology and setup, followed by a number of experiments that expose the core benefits of our IMCF framework and its internal EP algorithm compared to baseline techniques. It also carries out a control study for parameters of the EP algorithm and concludes with an Energy Conservation Study.

A. Methodology

This section provides details regarding the algorithms, metrics and datasets used for evaluating the performance of the proposed approach.

Testbed: Our evaluation is carried out on our laboratory VMware private datacenter. Our computing node comprises of a Ubuntu 18.04 server image, featuring 8GB of RAM with 2 virtual CPUs (@ 2.40GHz). The image utilizes fast local 10K RPM RAID-5 LSILogic SCSI disks, formatted with VMFS 6 (1MB block size).

Datasets: We have adopted a trace-driven experimental methodology in which real datasets are fed into our simulator executed on the testbed. This allows repeatable execution of

TABLE II

META-RULE TABLE (MRT) FOR FLAT EXPERIMENTS

Description	Time/Duration	Action	
Night Heat	01:00 - 07:00	Set Temperature	25
Morning Lights	04:00 - 09:00	Set Light	40
Day Heat	08:00 - 16:00	Set Temperature	22
Midday Lights	10:00 - 17:00	Set Light	30
Afternoon Preheat	17:00 - 24:00	Set Temperature	24
Cosmetic Lights	18:00 - 24:00	Set Light	40
Energy Flat	for three years	Set kWh Limit	11000
Energy House	for three years	Set kWh Limit	25500
Energy Dorms	for three years	Set kWh Limit	480000

TABLE III

IFTTT CONFIGURATIONS FOR FLAT EXPERIMENT

IF	THIS	THEN	THAT
Season	Summer	Set Temperature	25
Season	Winter	Set Temperature	20
Weather	Sunny	Set Temperature	20
Weather	Cloudy	Set Temperature	22
Weather	Sunny	Set Light	0
Weather	Cloudy	Set Light	40
Temperature	>30	Set Temperature	23
Temperature	<10	Set Temperature	24
Light Level	>15	Set Light	9
Door	Open	Set Light	0

workloads under different control parameters. More specifically, we utilize anonymized measurements from a real residential apartment that comprises of a variety of sensors, submeters and $\approx$ 5,668,878 readings (1.09 GB in total). These are real datasets of residential data collected by the "Center for Advanced Studies in Adaptive Systems" (CASAS) [12] at Washington State University. CASAS serves to meet research needs around testing of the technologies using real data through the use of a smart-home environment located on the WSU Pullman campus.

- **Temperature Dataset:** The 700 MB dataset contains 3,555,238 readings on a second basis between October 2013 and December 2016. The readings, which are recorded at a residential apartment of a volunteer adult, include temperature and door/window sensor data.
- **Light Dataset:** The 416 MB dataset contains 2,113,640 readings on a second basis between October 2013 and December 2016. The readings include the light data.

To evaluate the scalability of our propositions for residential buildings of various scales, we have generated three realistic datasets by replicating the above onto various building sizes. The resulting datasets are the following:

- **Flat Dataset:** A single user flat/apartment dataset consisted of one bedroom, a bathroom and a kitchen. The apartment has a single split unit to warm/cool an area size of 50 m^2. It has a size of 1.09 GB.
- **House Dataset:** A residential house dataset generated by replicating, mixing up the readings and multiplying the real dataset by a factor of four. It has three bedrooms and four split units used by four residents. The area size is $\approx$200 m^2. It has a size of 4.50 GBs.
- **Dorms Dataset:** A University Campus dataset (dorms) generated synthetically from the initial datasets. We have generated 50 dorm apartments consisting of two bedrooms (10 m^2 / room) with a shared bathroom, a kitchen and two split units. The total area size of the dorms is $\approx$ 2000 m^2 and has a size of 20 GBs.

Metrics: Our cost metrics for each meta-rule (MR_i) are defined in Section II and summarizes as follows:

- **Convenience Error** $ce_j(MR_i)$ *is the difference between the desired output value and the actual value.*
- **Energy Consumption** $e_j(MR_i)$ *is the energy consumption of device* j.
- **CPU Time** (F_T) *is the processing time required by the controller for running the optimization function and calculating the output for all meta-rules.*

The mean and standard deviation of the results is shown with error bars in all experimental studies that follow, based on ten repetitions.

Algorithms: Here we provide a concise overview of the compared methods and algorithms considering the Meta-Rule-Table (MRT), which is inspired from real preferences recorded by users.

- **Baseline Approaches (NR, MR):** the two baseline approaches are explained in Section II. The No-Rule (NR) approach ignores all rules in the Meta-Rule-Table and does not modify the behavior of the autonomous devices. The Meta-Rule (MR) method ignores the energy consumption and executes all rules greedily in the Meta-Rule-Table for satisfying all meta-rules set by the user.
- **If-This-Then-That (IFTTT):** this method executes the IFTTT preferences (see Table III) used in the flat dataset. The dataset was collected from the official IFTTT website. For the evaluation we measure F_{CE}, i.e., percentage of convenience a user will get from executing the IFTTT rules against all rules (recorded in the MRT table).
- **Energy Planner (EP) algorithm:** For the construction of the EP algorithm we have set the number of rules activation/deactivation in each iteration (k), a savings percentage amount (s), and the number of iterations (τ_{max}) and detailed evaluation follows for these parameters in sub-sections III-C and III-D.

B. Performance Evaluation

In this experimental series, we evaluate the performance of the proposed EP framework against all algorithms over all datasets introduced, with respect to Energy Consumption and the Convenience Error. Figure 6 demonstrates the trade-off between the Energy Consumption (F_E), the Convenience Error (F_{CE}) and the CPU Execution Time (F_T) between all approaches. The NR approach obtained the worst F_{CE} = 62% of the whole dataset, and the best F_E = 0 *kWh*. The EP algorithm obtained an impressive F_{CE} of around 2%-4% and the second lowest F_E. The IFTTT & MR algorithms are greedy in regards of Energy Consumption, thus their kWh consumed are very high. The main difference between the

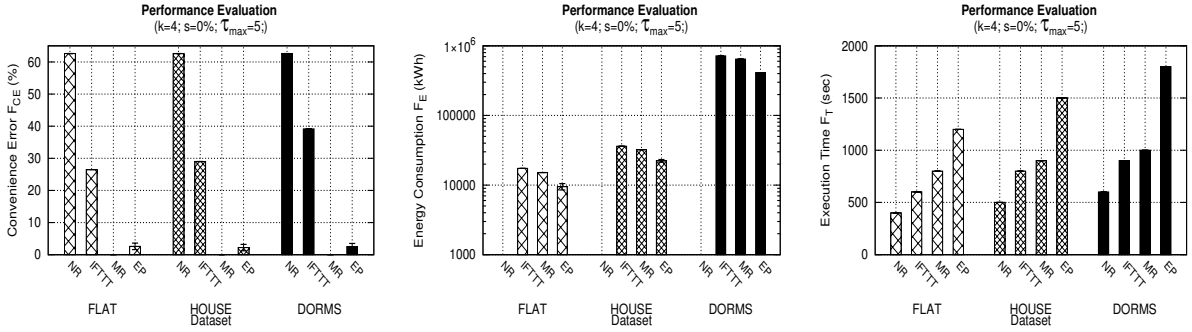

Fig. 6. **Performance Evaluation:** Evaluation in terms of the Convenience Error (F_{CE}), the *Energy Consumption* (F_E) and the CPU Execution Time (F_T) in all datasets.

two is that IFTTT has F_{CE} = 26% in the residential flat case, F_{CE} = 29% in the case of a house, and F_{CE} = 39% in the dorms case, while the MR satisfies all the meta-rules, thus its F_{CE} is 0%, which is the best possible obtained.

In the residential flat case, the preferred energy budget was configured to 11000 *kWh* for all three years, and the EP managed to save up to 10% of energy, which is approximately 9500 *kWh*, with a reasonable F_{CE} around 2%-3%. In the case of a house, the preferred energy budget was configured to 25500 *kWh* for all three years, and the EP managed to achieve approximately 22300 *kWh*, with F_{CE} around 2%-2.5%. In the dorms case, the preferred energy budget was configured to 480,000 *kWh* for all three years, and the EP managed to achieve approximately 410,000 *kWh*, with a reasonable F_{CE} around 2.5%-3%. Here it is important to notice, that the difference between the MR and the EP in terms of energy consumption, is relatively high and particularly ≈5,000kWh for the flat dataset, ≈10,000kWh for the house dataset, and ≈150,000kWh for the dorms dataset.

The fastest execution time was achieved by NR since it simply executes an error calculation ignoring all rules. The EP hill climbing approach, on the other hand, searches the decision space for a solution that will optimize the user's convenience and satisfy the energy constraint, at the same time, which is a much more time-consuming process. Finally, the MR greedy approach focuses only on minimizing the Convenience Error, which means executing all meta-rules without any iterative processes or calculations, since F_{CE}=0%.

C. k-opt Evaluation

In the second experiment, we evaluate the performance of the proposed EP framework against different ks (rule modifications), with respect to Energy Consumption and the Convenience Error. Figure 7 illustrates that by using four activation/deactivation rule modifications in each iteration we obtain the best F_{CE}. The worst F_{CE} occurred when we used two rule modifications. In the residential flat case, the energy consumed was in every case approximately the same and around 9500 *kWh*. What actually made a difference was the F_{CE}, which decreased from 3.3% to 2.6% in the flat case, from 3.0% to 2.2% in the house scenario, and from 3.4% to 2.5% in the dorms scenario, as we increased the activation/deactivation rule modifications in each iteration.

This is due to the hill climbing approach performing bigger "jumps" towards the local optimum at each step and thus searching the solution space more effectively. As the number of k rule modifications increases, the F_{CE} is decreasing gradually while the F_E is approximately at the same level.

D. Initialization Evaluation

In the third experimental series, we evaluate the performance of the proposed EP framework using different initialization strategies, with respect to Energy Consumption (F_E) and the Convenience Error (F_{CE}). In the first (all-1s) case, we have initially activated and applied all rules. In the second (random) case, we have uniformly randomly activated some rules and in the last (all-0s) case, we have initially deactivated all rules. Figure 8 presents the F_{CE} that increases by using the "all-deactivated" (i.e., all-0s) rules strategy, hence consuming less energy and in contrast to the "all-activated" (i.e., all-1s) and the "random" rule strategies.

In the residential flat case, starting from all-1s, moving to random and finally to all-0s, we observe an increase on the F_{CE} from approximately 2.6% to 3.1%, but respectively there is a decrease on the F_E from approximately 9500 *kWh* to 8600 *kWh*. In the house scenario, starting from all-1s, moving to random and finally to all-0s, we observe an increase on the F_{CE} from approximately 2.2% to 2.7%, but respectively there is a decrease on the F_E from approximately 22300 *kWh* to 20500 *kWh*. In the dorms case, starting from all-1s, moving to random and finally to all-0s, we observe an increase on the F_{CE} from approximately 2.5% to 3.0%, but respectively there is a decrease on the F_E from approximately 410,000 *kWh* to 382,000 *kWh*. This is due to the hill climbing approach that needs to perform more iterations in the solution space to find the local optimum, and consequently an optimal energy plan, when all rules are deactivated.

E. Energy Conservation Study

In the fourth experimental series, we evaluate the performance of the proposed EP approach over various savings percentages, with respect to Energy Consumption and Convenience Error. This evaluation is inspired by the SAVES is

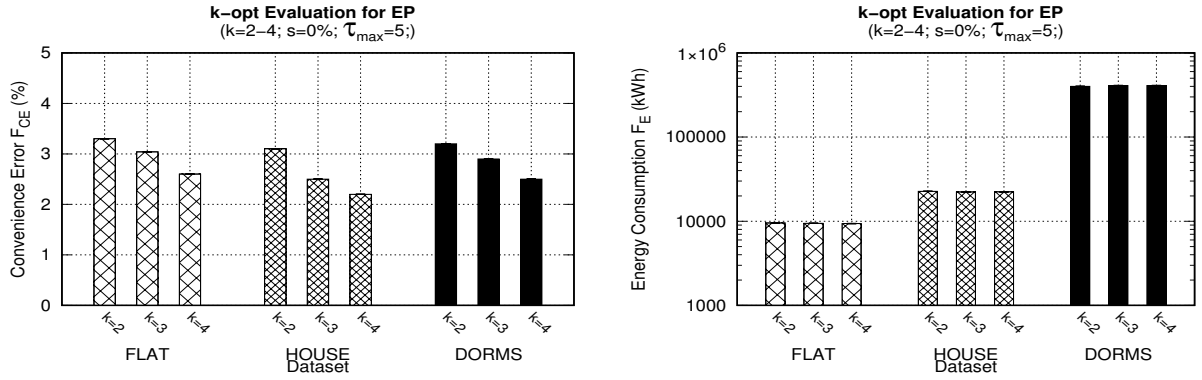

Fig. 7. **k-opt Evaluation:** Evaluation in terms of the Convenience Error (F_{CE}) and the *Energy Consumption* (F_E) based on the number of modified rules (activated/deactivated), in all datasets.

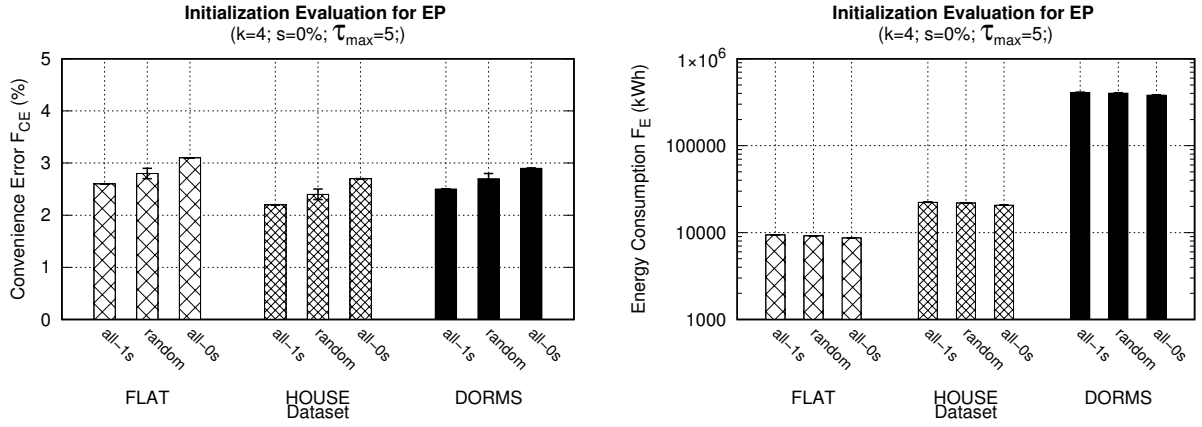

Fig. 8. **Initialization Evaluation:** Evaluation in terms of the Convenience Error (F_{CE}) and the *Energy Consumption* (F_E) based on various initialization techniques, in all datasets.

TABLE IV
EVALUATING OUR SYSTEM PROTOTYPE WITH RESPECT TO ENERGY CONSUMPTION (F_E) AND CONVENIENCE ERROR (F_{CE})

Time Duration	Energy Consumption (F_E)	Convenience Error (F_{CE})
Week	130.64 kWh	2.35%

TABLE V
INDIVIDUAL RESIDENT CONVENIENCE ERROR (F_{CE})

Users	Convenience Error (F_{CE})
Father	0.8006%
Mother	0.7899%
Daughter	0.7595%

an inter-dormitory energy-saving competition that took place on 2014 - 2016 and that we outlined in the introduction. SAVES aimed at delivering 8% average electricity savings in participating dormitories.

Figure 9 shows that by increasing the potential energy savings there is a slight increase on the F_{CE} clearly demonstrating the trade-off between those two objectives. The trade-off ranges between 5-40% of energy savings (that is around 1500 *kWh* in the residential flat case) for 1-3% increase on the F_{CE} can be considered as a fair exchange.

F. Prototype Evaluation

In the final experimental series, we deployed an instance of our real prototype system for a family of three persons for one week. Particularly, we allowed each person to configure their personal preferences using the Mobile APP that interacts with an IMCF-LC node on a Linux VM on our datacenter described earlier. Particularly, each individual resident entered approximately three different meta-rules according to their personal preferences. One of them have set the weekly energy consumption (kWh) limit to 165kWh. This results in configuration data of approximately 65 bytes / user stored in the MariaDB persistency layer. In order to measure the environmental parameters (i.e., temperature, light) we use data from the open weather API. We measure again the performance of the proposed EP framework in regards to Energy Consumption and Convenience Error.

The F_E and F_{CE} results for our evaluation using real Weather Forecast data are summarized in Table IV. In respect to F_{CE} our observation is that EP is indeed an efficient approach for retrieving great user satisfaction, as it performs

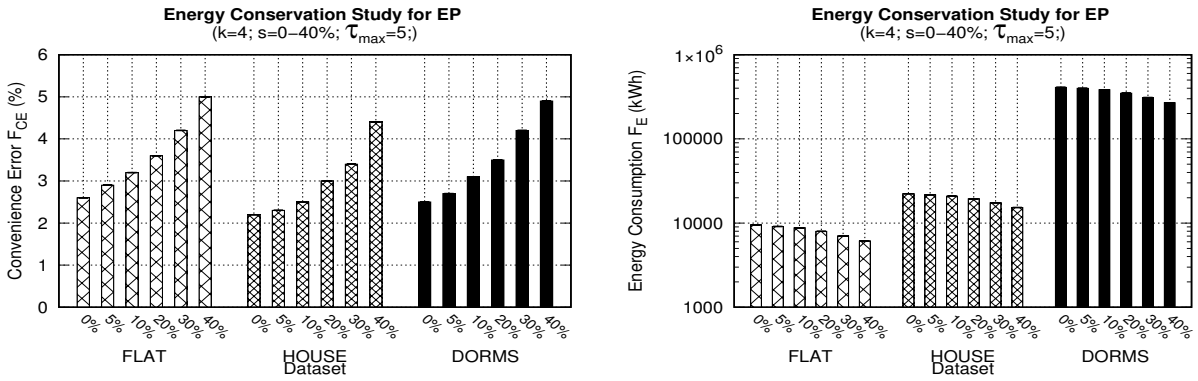

Fig. 9. **Energy Conservation Study:** Evaluation in terms of the Convenience Error (F_{CE}) and the *Energy Consumption* (F_E) based on different saving values, in all datasets.

in 4 seconds on average with an Average Convenience Error $\approx$ 2.35%. Table V demonstrates for each individual resident their own Average Convenience Error values in respect with their configured meta-rules, showing both a consistent and high satisfaction close to 99.7% for all residents. Another observation is that Energy Consumption $\approx$ 130.64 kWh is within the preferred budget limit as pre-configured by the user, and the system behaves correspondingly to what we observed in the simulations.

IV. Related Work

In this section, we provide the related work using a breadth-to-depth approach that aims to provide additional pointers to the interested reader.

A. IoT Data Management

The uptake of IoT in recent years has brought a revived interest on data management and data engineering solutions, architectures and applications with a focus on data ingestion [13], analytic architectures for streaming data [14] as well as relevant benchmarking [15]. From the application perspective, a specific focus has been given to privacy [16], context awareness [17], temporal analytics [18], localization [19], [20] and telco big data [21].

Green Data Management has been a complementary and related topic with intensive research over the years, particularly in data center [22] and data warehouse design [23], green-aware route planning in GIS systems [24], but the focus on smart IoT actuation application frameworks has been overlooked over the years. Our study aims to put a focus on smart energy systems and smart actuations that aims to contribute in curbing the CO_2 increase of IoT from 4% to 8% by 2025.

B. Smart Energy Management Systems

In this subsection we overview energy management systems for three different contexts, stemming both from the industrial and academic sectors.

Photovoltaic Home Energy Management: The Sunny Home Manager [25] (HM) controllers by SMA monitors power flows, particularly the production of AC power from the inverters and the consumption of AC power from the households (recorded by an energy meter). HM then manages the power consumption workloads accordingly (e.g., when to operate a washing machine or smart car charger so that solar energy self-consumption is optimized). This is achieved with its open *Simple Energy Management Protocol (SEMP)* or the industry-wide adopted EEBUS [26] protocols with its KEO reference implementation. However, these protocols are geared for load management inside smart buildings rather than for enabling users achieve some long-term energy (energy consumption) targets as we do in our work. As such, these energy home managers have a complementary role to the energy planning propositions we present in this paper.

Smart Thermostats: The Nest.com Learning Thermostat is a programmable and self-learning Wi-Fi-enabled thermostat that optimizes cooling and heating to conserve energy. However, there are the following differences with IMCF: (i) these thermostats do not enable the adaptation of convenience preferences to meet the long-term energy planning targets of individuals or group of individuals (see examples in Section I); and (ii) these require learning data from users (e.g., location) that might be a privacy concern. Similarly, prior research [27], [28], was mainly concerned with improving comfort levels of HVAC system but not long-term energy planning targets.

C. Rule Automation Workflows (RAW)

In this subsection we cover complementary work of RAW pipelines and the competing approaches to achieve the exploration of the RAW search space.

Real-time IFTTT: Heo et al. [29] implemented RT-IFTTT, a real-time IoT language and its framework that uses trigger condition-aware flexible sensor polling intervals. The RT-IFTTT language extends the existing IFTTT syntax, and allows users to specify real-time constraints for their applets. Again, this system doesn't enable long term energy planning.

RAW Informed Search Methods: are generally characterized by a *utility* in scanning the solution space to reach the goal

these. These algorithms utilize some kind of an evaluation function that greedily assesses some distance of the current solution from a target (e.g., in the case of *A* heuristic* search). Unfortunately, A*-search always requires some evaluation function that is not available in our case as we really do not know the convenience target of a user within the agreed energy budget. As such, we have to rely on stochastic informed search algorithms (e.g., *simulated annealing* and *hill climbing*), which probabilistically carry out a similar task but without requiring a rigid target function. The EP algorithm proposed in this work, is founded on *hill climbing* space exploration method that deploys a user-controlled energy amortization strategy and domain heuristics to bring forward the expected result.

V. Conclusions and Future Work

In this paper, we present the IoT Meta-Control Firewall (IMCF), an innovative system architecture and application, which internally deploys an AI-inspired Energy-Planner (EP) algorithm that exploits domain-specific operators to balance the trade-off between convenience and energy consumption while managing RAW pipelines of users. Our experimental evaluation, with extensive real traces from an apartment, a house and a campus, show that a user can have 10% energy savings at only 2-4% decrease in the user's convenience. We also found that the execution of EP is fairly fast, carrying out the computation in about 4 seconds for the largest datasets. Given that our approach requires no training data and only a primitive preference profile, this can be easily integrated in actuation platforms, as we have demonstrated with IMCF.

In the future, we plan to further investigate multiple energy planners with conflicting interests but also to investigate the so-called IMCF-Cloud extensions that will enable IMCF to operate as a CMC controller in the cloud. We also aim to look at CO2 reductions methods with algorithms geared towards the environment. Finally, we aim to investigate power workload identification methods for power-hungry devices (e.g., white devices, electric vehicles, heating) and how to reschedule those workloads in a environmental friendly manner.

References

[1] J. Chen, Y. Chen, Z. Chen, A. Ghazal, G. Li, S. Li, W. Ou, Y. Sun, M. Zhang, and M. Zhou, "Data management at huawei: Recent accomplishments and future challenges," in *35th IEEE International Conference on Data Engineering*, pp. 13–24, 2019.

[2] L. Yao, Q. Z. Sheng, and S. Dustdar, "Web-based management of the internet of things," *in IEEE Internet Computing, vol. 19, iss. 4, pp. 60–67*, 2015.

[3] A. A. Fuqaha, M. Guizani, M. Mohammadi, M. Aledhari, and M. Ayyash, "Internet of things: A survey on enabling technologies, protocols, and applications," *IEEE Comm. Surv. Tutor., vol. 17, no. 4, pp. 2347–2376*, 2015.

[4] S. Li, L. D. Xu, and S. Zhao, "The internet of things: a survey," *Information Systems Frontiers*, vol. 17, no. 2, pp. 243–259, 2015.

[5] B. Ur, M. P. Y. Ho, S. Brawner, J. Lee, S. Mennicken, N. Picard, D. Schulze, and M. L. Littman, "Trigger-action programming in the wild: An analysis of 200,000 ifttt recipes." *In ACM CHI Conference on Human Factors in Computing Systems*, pp. 3227–3231, 2016.

[6] X. Mi, F. Qian, Y. Zhang, and X. Wang, "An empirical characterization of ifttt: Ecosystem, usage, and performance," in *Internet Measurement Conference*, pp. 398–404, 2017.

[7] W. Brackenbury, A. Deora, J. Ritchey, J. Vallee, W. He, G. Wang, M. L. Littman, and B. Ur, "How users interpret bugs in trigger-action programming," in *ACM CHI Conference on Human Factors in Computing Systems*, pp. 1–12, 2019. [Online]. Available: https://doi.org/10.1145/3290605.3300782

[8] "Erasmus student network (esn)," *https://esn.org/saves*, 2014.

[9] A. Tongaonkar, N. Inamdar, and R. Sekar, "Inferring higher level policies from firewall rules," in *21th USENIX Large Installation System Administration Conference*, pp. 17–26, 2007. [Online]. Available: http://www.usenix.org/events/lisa07/tech/tongaonkar.html

[10] "Open home automation bus," *https://www.openhab.org/*, 2010.

[11] "Nginx web server," *https://www.nginx.com/*, 2004.

[12] A. Brown, "Washington state university," *in ACM Interactions*, vol. 9, no. 2, p. 51–53, 2002.

[13] C. Costa, G. Chatzimilioudis, D. Zeinalipour-Yazti, and M. F. Mokbel, "Efficient exploration of telco big data with compression and decaying," in *33rd IEEE International Conference on Data Engineering*, pp. 1332–1343, 2017.

[14] B. Del Monte, S. Zeuch, T. Rabl, and V. Markl, "Rhino: Efficient management of very large distributed state for stream processing engines," in *ACM SIGMOD International Conference on Management of Data*, pp. 2471–2486, 2020. [Online]. Available: https://doi.org/10.1145/3318464.3389723

[15] M. Poess, R. Nambiar, K. Kulkarni, C. Narasimhadevara, T. Rabl, and H. Jacobsen, "Analysis of tpcx-iot: The first industry standard benchmark for iot gateway systems," in *34th IEEE International Conference on Data Engineering*, pp. 1519–1530, 2018.

[16] S. Ghayyur, Y. Chen, R. Yus, A. Machanavajjhala, M. Hay, G. Miklau, and S. Mehrotra, "Iot-detective: Analyzing iot data under differential privacy," in *ACM SIGMOD International Conference on Management of Data*, pp. 1725–1728, 2018.

[17] D. Lohani and D. Acharya, "Smartvent: A context aware iot system to measure indoor air quality and ventilation rate," in *17th IEEE International Conference on Mobile Data Management*, vol. 2, pp. 64–69, 2016.

[18] A. Gal, A. Senderovich, and M. Weidlich, "Online temporal analysis of complex systems using iot data sensing," in *34th IEEE International Conference on Data Engineering*, pp. 1727–1730, 2018.

[19] A. Konstantinidis, P. Irakleous, Z. Georgiou, D. Zeinalipour-Yazti, and P. K. Chrysanthis, "Iot data prefetching in indoor navigation soas," *in ACM Trans. Internet Technol.*, vol. 19, no. 1, 2018.

[20] P. Mpeis, T. Roussel, M. Kumar, C. Costa, C. Laoudias, D. Capot-Ray, and D. Zeinalipour-Yazti, "The anyplace 4.0 iot localization architecture," in *21st IEEE International Conference on Mobile Data Management*, pp. 218–225, 2020. [Online]. Available: https://doi.org/10.1109/MDM48529.2020.00045

[21] C. Costa and D. Zeinalipour-Yazti, "Telco big data research and open problems," in *35th IEEE International Conference on Data Engineering*, pp. 2056–2059, 2019.

[22] B. Aksanli, J. Venkatesh, L. Zhang, and T. Rosing, "Utilizing green energy prediction to schedule mixed batch and service jobs in data centers," in *4th ACM HotPower Workshop on Power-Aware Computing and Systems*, 2011. [Online]. Available: https://doi.org/10.1145/2039252.2039257

[23] M. Poess and R. O. Nambiar, "Tuning servers, storage and database for energy efficient data warehouses," in *26th IEEE International Conference on Data Engineering*, pp. 1006–1017, 2010.

[24] J. Hu, B. Yang, C. S. Jensen, and Y. Ma, "Enabling time-dependent uncertain eco-weights for road networks," *Geoinformatica*, vol. 21, no. 1, pp. 57–88, 2017. [Online]. Available: https://doi.org/10.1007/s10707-016-0272-z

[25] "Sunny home manager 2.0," *https://www.sma.de/en/products/monitoring-control/sunny-home-manager-20.html*, 2019.

[26] L. Blume, "Sma is opening its doors to third-party suppliers," *http://tiny.cc/4jejjz*, 2015.

[27] D. Petrov, R. Alseghayer, P. K. Chrysanthis, and D. Mosse, "Smart room-by-room hvac scheduling for residential savings and comfort," in *The 10th International Green and Sustainable Computing Conference*, pp. 1–7, 2019.

[28] D. Petrov, R. Alseghayer, D. Mosse, and P. K. Chrysanthis, "Data-driven user-aware hvac scheduling," in *The 9th International Green and Sustainable Computing Conference*, pp. 1–8, 2018.

[29] S. Heo, S. Song, J. Kim, and H. Kim, "Rt-ifttt: Real-time iot framework with trigger condition-aware flexible polling intervals," in *IEEE Real-Time Systems Symposium*, pp. 266–276, 2017.